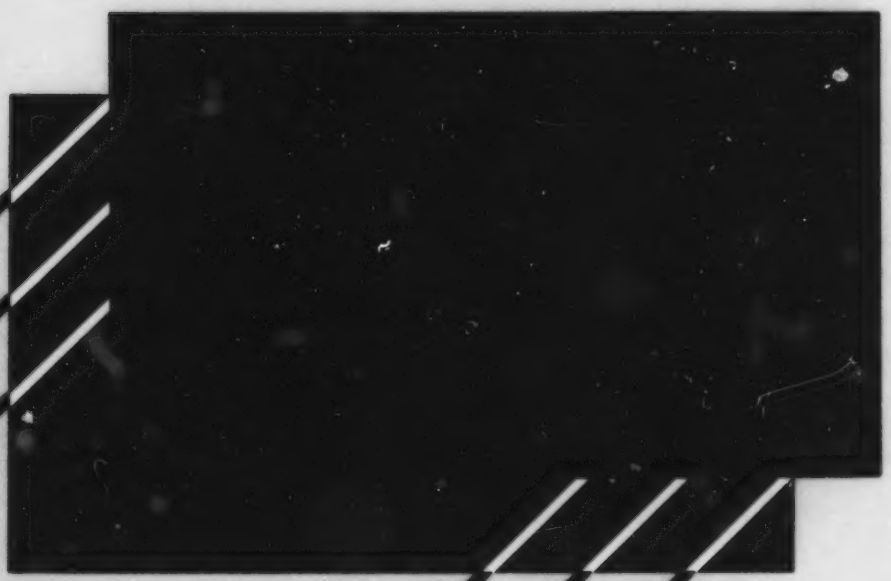
Gouvernement du Canada
Ministère des Communications

Government of Canada
Department of Communications

Le Centre canadien de recherche sur l'informatisation du travail
Canadian Workplace Automation Research Centre



Canada

SUN-TULIP USER GUIDE

by

Ron Ferguson

This report is the result of research done at the
Canadian Workplace Automation Research Center
under the direction of Joël Muzard, Expert Systems Group.

Department of Communications

Laval

November 1989

92-05716/

Nº de cat. Co 28-1/41-1989E
ISBN 0-662-17472-0

The opinions expressed are those of the author only.

* Une version française de ce document est aussi disponible.

Table of contents

Introduction 3

Levels of Comprehension 4
- Conceptual Comprehension 4
- Syntactic Comprehension 5
- Word Matching 5
- Limitations of Tulip's Conceptual Comprehension 5

Entering and Exiting Tulip 6

Implementation of the help function of Tulip system 7

Aborting Tulip's Execution 7

Input to Tulip 8
- Commands 10
- The Form of English Assertions Accepted 11
- The Form of English Queries Accepted 11
- Answering Queries by Word Matching 12

Formulation of the statements
- Conjunction 13
- Noun Phrases 13
- Determiners 14
- Pronouns 16
- Verbs 16
- Names 17

Dates and Times 18

Arithmetic Operations 18

Error Messages 18

French Text 18

Expressing Knowledge in Tulip 19

Introduction

TULIP (Text Understanding and Logical Inferencing Processor) is a natural language interface to an 'electronic scratchpad'. The user can make assertions and ask questions of the system in a subset of English. The user can also enter rules describing relationships between entities.

In addition to knowledge which the user supplies, Tulip has built-in expertise in certain areas. Specifically, Tulip is able to interpret business cards, dates and times, and certain arithmetic operations.

This user's manual describes the operation of Tulip on Sun Workstations. It is assumed that the user has a basic familiarity with Sun Workstations.

In this user's manual, example are given of input which Tulip does and does NOT accept. By an abuse of a standard convention in linguistics, input which is unacceptable to Tulip will be preceded by [*].

[*] Green ideas sleep furiously.

Levels of Comprehension

Tulip can be said to 'understand' English, in that it is able to use assertions and rules to find responses to queries. However the user should not expect Tulip to understand English to the extent that a human understands it. In order for a computer program to understand a natural language (such as English) at the human level, the program would have to have the experience, knowledge, and inferencing capabilities that a human has. No currently existing program has these capabilities.

In order to degrade gracefully, Tulip has three levels of comprehension. If Tulip is unable to understand a statement at the highest level of comprehension, it tries the next level down. The levels of comprehension from highest to lowest are :

[1] Conceptual comprehension.

[2] Syntactic comprehension.

[3] Word matching.

Conceptual Comprehension

At this level of comprehension, Tulip is able to relate different ways of saying the same thing, even when different words are used. For instance, Tulip assumes that the following assertions mean the same thing :

> Mr. John Smith works for Kentek.
> Kentek employs Mr. John Smith.

Thus Tulip is aware of a relationship between the words 'works for' and 'employs'.

Syntactic Comprehension

At this level of comprehension, Tulip is able to parse a sentence and is able to relate sentences which say the same thing using different syntactic forms. For instance, the following sentences give Tulip the same information :

Kentek employs Mr. John Smith.
Mr. John Smith is employed by Kentek.

In the above example, the second sentence is the passive form of the first sentence. Tulip also understands different ways of expressing the possessive :

the father of Jack
Jack's father

Tulip can also relate queries to assertions if the same expressions are used in both :

Kentek employs Mr. John Smith.
Who is Mr. John Smith employed by?

Word Matching

If a question is input which Tulip is unable to parse, Tulip finds those previously entered assertion(s) which contain the most words in common with the question. These assertions are displayed as possible answers to the query. Often this simple technique is able to retrieve the information the user wants, although irrelevant information may also be retrieved.

Limitations of Tulip's Conceptual Comprehension

Tulip has limited built-in conceptual comprehension. It understands most user input at the syntactic level rather than the conceptual level. Therefore, the user should attempt to employ a consistent terminology when using Tulip. The user can extend Tulip's conceptual comprehension by entering rules which define the relationships between words.

Entering and Exiting Tulip

To use Tulip you must be in the following directory :

/home/condor2/tulip

Load Tulip by typing :

tulip.bin

Tulip is a large program so it will take some time for the program to load. Once Tulip has loaded the following Quintus Prolog prompt will appear :

| ?-

Enter Tulip by typing :

tulip.

When Tulip is ready for user input, the following prompt appears :

Enter a sentence :

To exit from Tulip type :

exit. (or) quit.

This will return you to the Quintus Prolog prompt :

| ?-

To exit from Quintus Prolog type :

halt.

Implementation of the help function of Tulip system

To implement the help function while your in Tulip, type :

help.

Aborting Tulip's Execution

The user can abort Tulip while it is executing by typing ^C (control-C). The following prompt is displayed :

Prolog interruption (h for help)?

To abort from Tulip type :

a

This will put one in Quintus Prolog. One can re-enter Tulip by typing :

tulip :

Input to Tulip

The input to Tulip consists of the following basic types :

[1] Commands - A command is a line of input which begins with a keyword (eg. read, save, restart) and ends with a period followed by a carriage return. They are used to tell Tulip to perform special functions. An example of a command is :

restart.

[2] Assertions - An assertion is a line of english input which makes a statement of fact and ends with a period followed by a carriage return. Tulip remembers assertions and uses them to answer queries. An example of an assertion is :

Bill is the husband of Mary.
If the first word in a sentence is not a proper name, it is not necessary to capitalize it.

a cat is on the mat.

[3] Queries - A query is a line of english input in the form of a question which ends with a question mark followed by a carriage return. When the user enters a query, Tulip tries to find an answer to the query based on what it has been told. An example of a query is :

who is Mary's husband?

It is not necessary to capitalize the first word of a query.

[4] Rules - A rule is a line of English-like input of the form :

<Conclusion> **if** <Premise1> (**and** <Premise2>...).

where <Conclusion> and the <Premise>s are simple statements which may contain 'variables' which act as standin's for objects. A variable is a capitalized letter other than 'A' or 'I'. Rules are used to establish relationships between concepts. An example of a rule is :

X is the wife of Y if Y is the husband of X.

Using this rule and the assertion given above that :

Bill is the husband of Mary.

Tulip can answer the question :

Who is Bill's wife?

The meaning of the verbs 'have' and 'is' depends on the type of object the verb takes. As a result, a variable can NOT be used as the object of the verbs 'have' or 'is' :

[*] X has Y if X owns Y.
[*] X is a Y if the type of X is Y.

However, a variable can be used as the subject of the verbs 'have' or 'is' :

X has a window office if X is a manager.
X is a mammal if X has fur.

Often, a rule which involves just one variable can also be expressed as an assertion. For example the preceding rules can be expressed as :

Every manager has a window office.
Everything that has fur is mammal.

[5] Business Cards - If a line is entered which does not end with a period or question mark, Tulip assumes that it is the first line of a business card. If the first line of a business card happens to end with a period, a space should be typed after the period and before the carriage return. Tulip displays the following prompt to indicate that it expects more lines of input for the business card :

I :

A blank line terminates the business card input.

Commands

save <name> .

- saves the sentences the user has typed in since the last Save or Restart command. The sentences are saved as <name>, where <name> is a alphanumeric word. If <name> has previously been used in a save command, the sentences are added to the end of <name>. Sentences which are saved can be read back in using the Read command.

read <name> .

- reads in the sentences which were saved using 'Save <name>.' Note : The 'save <name>.' command saves sentences in a file called '<name>.text' in Tulip's 'knowledge' directory. It is possible to put a text file in the knowledge directory and read it using the 'read' command. Also, the knowledge files can edited using an editor.

restart.

- Erases Tulip's memory

debug.

- displays Tulip's processing of a sentence. This is very detailed and will probably not be useful to the average user.

end debug.

- turns off the detailed display of the processing of sentences.

help.

- explains how to use Tulip.

The Form of English Assertions Accepted

In general, Tulip accepts English statements which express a literal statement of fact.

Negative information can not be expressed. Thus one can not enter a statement with the word 'not' in it. eg.

[*] George is not a programmer.

Statements with the word 'or' are also not accepted. eg.

[*] Helen or Mary is married to Jim.

An assertion must begin with a noun phrase which is the subject of the statement.

The Form of English Queries Accepted

Tulip accepts both Wh and yes/no questions. A Wh question is a question which begins with a question word such as 'who', 'what', 'which', 'where'. 'How many' is also considered to be a Wh word. The Wh word can be preceded by a preposition. The following are examples of Wh questions :

Who is Frank's wife?
To whom is Frank married?
How many people work for Kentek?
Which employees live in Laval?

Note that the wh word must be the first word (or second word if it is the object of a preposition) in the sentence. Thus the following query is not allowed :

[*] Jack is married to whom?

The above query should be expressed as :

To whom is Jack married?
or
Whom is Jack married to?

The word 'how' by itself is not handled.

Jack runs quickly.
[*] How does Jack run?

A *yes/no* question is a question which can be answered by 'yes' or 'no'. Such questions usually begin with the verb 'is' or 'does' :

Does Frank work for Kentek?
Is Frank a manager?

Answering Queries by Word Matching

If a question is input which Tulip is unable to parse, Tulip uses word matching to try to find relevant information. Thus if one enters keyword(s) followed by a question mark, Tulip will display those assertions which contain the largest number of keywords.

Conjunction

Only certain uses of the word 'and' are allowed. One can use 'and' between statements :

Joe lives in Toronto and Henry lives in Montreal.

If the subject is the same in both statements, it can be omitted in the second statements :

Joe lives in Guelph and works in Toronto.

Relative clauses (see noun Phrases) can also be joined by 'and'.

Noun Phrases

A noun phrase can be either a common noun phrase, eg.

the tall man

or a proper name, eg.

John Doe

The words in a proper name must be capitalized.

A common noun phrase (but not a proper noun phrase) can be modified by a (conjunction of) relative clause(s). For instance, the following is a noun phrase.

the man who lives in Halifax and is employed by Acme Inc.

Tulip does not handle reduced relatives, such as :

[*] the man employed by Acme Inc.

This should be written as :

the man who is employed by Acme Inc.

Common noun phrases can be either singular or plural :

the cat
the cats

However, conjunctions of noun phrases are not allowed. For example,

[*] the men and women
[*] Bill and Jane

are not allowed.

A list of words inside double quotes is treated as a singular proper noun.

"The Young and the Restless" is a television program.

Determiners

A common noun phrase typically begins with a determiner such as 'a', 'the', 'some', 'all', or 'every'. The determiners 'most' and 'few' are not allowed.

If an assertion begins with the determiner 'a', eg :

a bear has claws.

Tulip can not determine whether the user means :

some bear has claws or every bear has claws.

Therefore, Tulip displays a menu with these two choices and asks the user to choose the appropriate meaning.

If 'a' is used in any position other than the beginning of a sentence or is used meaning 'some' at the beginning of a sentence, Tulip assumes that a new object is being referred to. In contrast, if 'the' is used, Tulip assumes that the user is referring back to an object that was mentioned in a previous sentence. Thus, if the statement :

George has a car.

has been entered, and the user then enters :

Helen likes the car.

Tulip assumes that 'the car' refers to the last mentioned car, ie. George's car.

On the other hand, if the user had entered :

Helen likes a car.

Tulip would not assume that it was George's car that Helen likes.

The only determiner allowed with a proper noun is 'the' :

The Volvo
[*] a Volvo

Pronouns

Tulip will attempt to find the referent of third person singular pronouns such as 'he', 'she', 'him', 'her', and 'it'. It will also try to find a referent for prossessive third person singular pronouns such as 'her' and 'his'.

Jack works for Acme Inc.
He owns a car.
Alice is his wife.
She gave him his coat.
who owns a car?
who is Jack's wife?
what did Alice give to Jack?

In the above example, 'he', in the secound sentence, is assumed to refer to Jack, the last mentioned male.
Plural pronouns, such as 'they', 'them', and 'their', are NOT handled.

Verbs

Tulip accepts verbs in the present, past, or future tense. However, the tense of verbs is only used to interpret dates. Apart from that, the tense of a verb is ignored.

Modal verbs (verbs which modify other verbs) such as 'can', 'should', etc., are also accepted but are ignored.

Thus the following sentences are given the same meaning by Tulip :

Tweety can fly.
Tweety should fly.
Tweety could fly.
Tweety will fly.
Tweety flies.

Other than modal verbs, Tulip can not parse sentences which contain verbs that take other verbs as arguments (eg. persuade) :

[*] Tim persuaded Tom to hire Mary.

Names

Tulip is sometimes able to automatically determine what type of object a proper name refers to. Tulip knows common first names for men and women. Tulip uses this knowledge to try to class the people mentioned in a sentence as being men or women. Similarly, Tulip will assume that proper names which end with 'Limited', 'Incorporated', or 'Company', refer to organizations. Tulip also knows the names of cities, provinces, and states.

When a name appears in a sentence, Tulip tries to match the name with an object that it already knows about. The name does not have to exactly match the already established name of the object, but the new name must be a subset of the original name. For example, if Tulip knows about a person called 'Mr. Jack Smith' is being refered to. In the case where a name could refer to more than one person that Tulip knows about, the most recently mentioned person is chosen. Thus if 'Mr. Jack Smith Jones' has been mentioned more recently than 'Mr. Jack Jones'. When a person is first mentioned, his most complete name should be used since that individual can only be referred to by that name or subparts of that name from that point onwards.

Tulip does not allow an object to have more than one name.

Dates and Times

See the 'dates' demo for examples of Tulip's use of dates and times.

Arithmetic Operations

See the 'cars' demo for examples of Tulip's use of 'average', 'maximum', 'total', and 'minimum'.

Error Messages

If Tulip can not parse an assertion or query, it displays the following message :

Syntactic parser failed towards the end of this segment :

It then displays as much of the input sentence as it was able to parse. Usually the last word displayed in the sentence is the one that Tulip had trouble with.

If Tulip is able to parse a query but is not able to find a solution, Tulip asks the user if she wants a list of potentially relevant facts. If the user answers 'Yes', Tulip uses word matching to find those assertions which have the most words in common with the query.

French Text

Tulip has limited ability to handle French test. In general, Tulip is not able to parse French sentences. However Tulip can use word matching on certain French words. Its French vocabulary is limited to words which might appear on a business card and terms used in the field of artificial intelligence.

To use French accents, one must include the following line in one's .cshrc file :

```
setenv DEFAULT_FONT "/home/condor/critter/fonts/f_screen.r.14"
setenv EMACS_FONT "/home/condor/critter/fonts/f_serif.r.14"
```

and have the following lines in a .emacs_pro file in one's home directory :

```
(load  "/home/condor/critter/.emacs_pro")
(load  " /../isabelle/emacs/dead_key_accents")
```

French accents can be added to letters by using the keys marked R1 to R5 which appear to the right of the main keys on a Sun 3. The accents are added by typing the appropriate R key followed by the letter the accent is to be added to. The accents associated with the R keys are :

R1 - aigu
R2 - grave
R3 - cidille
R4 - trima
R5 - circonflexe

Expressing Knowledge in Tulip

Tulip can be used to express inheritance of properties between classes of objects. For instance, one can enter :

every bird can fly.
every robin is a bird.
Ernie is a robin.

If one then asks :

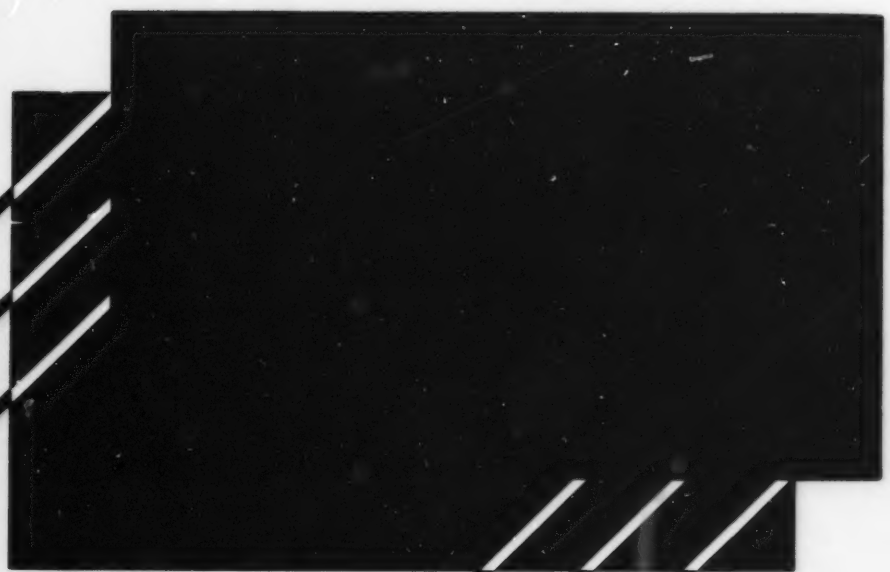
Can Ernie fly?

Tulip will respond 'Yes'. Since the user has stated that every robin is a bird, all robins will inherit the properties that are associated with birds (eg. flying).

Gouvernement du Canada
Ministère des Communications

Government of Canada
Department of Communications

Le Centre canadien de recherche sur l'informatisation du travail
Canadian Workplace Automation Research Centre



Canada

GUIDE DE L'UTILISATEUR DU SYSTÈME SUN-TULIP

par

Ron Ferguson

Ce rapport est le résultat de travaux de recherche effectués au Centre canadien de recherche sur l'informatisation du travail sous la direction de Joël Muzard, Groupe systèmes experts.

Ministère des Communications

Laval

Novembre 1989

Nº de cat. Co 28-1/41-1989F
ISBN 0-662-95735-0

Les opinions émises dans ce rapport n'engagent que l'auteur.

* This report is also available in English

Table des matières

Introduction....3

Niveaux de compréhension....4
- La compréhension conceptuelle....5
- La compréhension syntaxique....5
- L'association de mots....6
- Limite de la compréhension conceptuelle du système Tulip....7

Procédures d'entrée et de sortie....7

Mise en application de la fonction d'aide du système Tulip....8

Suspension de l'exécution du programme Tulip....9

Données d'entrée pour le système Tulip....10
- Les instructions....13
- Forme des assertions en anglais admissibles....14
- Forme des interrogations en anglais admissibles....15
- Réponse aux interrogations par appariement de mots....16

Formulation des énoncés :
- Conjonction....17
- Les syntagmes nominaux....17
- Les déterminants....19
- Les pronoms....21
- Les verbes....22
- Les noms....23

La date et l'heure....24

Les opérations arithmétiques....24

Les messages d'erreur....24

Les textes en français....25

L'expression des connaissances dans le système Tulip....26

Introduction

Le système TULIP (Text Understanding and Logical Inferencing Processor) est une interface de langage naturel pour un bloc-notes électronique. L'utilisateur peut se servir d'un sous-ensemble de la langue anglaise pour faire des assertions et poser des questions au système. Il peut également introduire dans le système des règles décrivant des relations entre des données d'entrée.

En plus des connaissances introduites par l'utilisateur, le système TULIP a une base de connaissances incorporée qui porte sur certains domaines : il peut interpréter les cartes professionnelles, la date et l'heure ainsi que certaines opérations arithmétiques.

Ce guide décrit le fonctionnement du système TULIP sur les postes de travail Sun et a été écrit en supposant que le lecteur connaît les rudiments des postes de travail Sun.

Dans ce guide, nous donnons des exemples d'informations admissibles et NON admissibles par le système TULIP. Par convention, les informations non admissibles par le système TULIP seront précédées de [*]. Exemple :

[*] Green ideas sleep furiously.

Niveaux de compréhension

On peut dire que le système TULIP comprend l'anglais en ce sens qu'il peut utiliser des assertions et des règles pour trouver des réponses à des interrogations. Toutefois, l'utilisateur ne peut s'attendre à ce que le système TULIP comprenne l'anglais aussi bien qu'un être humain. Pour qu'un programme machine comprenne un langage naturel (tel que l'anglais) aussi bien qu'un être humain, il faudrait qu'il ait l'expérience, les connaissances et les capacités d'inférence d'un être humain. A l'heure actuelle, ce n'est le cas d'aucun programme.

Le système TULIP a trois niveau de compréhension, ce qui permet d'obtenir une réduction progressive de la puissance de traitement. Si le système ne peut comprendre un énoncé au niveau de compréhension le plus élevé, il recommence l'opération au niveau inférieur immédiat. Par ordre de décroissance, les niveaux de compréhension sont les suivants :

[1] la compréhension conceptuelle;

[2] la compréhension syntaxique;

[3] l'association de mots.

La compréhension conceptuelle

A ce niveau de compréhension, le système TULIP peut faire la relation entre des façons différentes de dire la même chose, même lorsqu'on utilise des mots différents. Par exemple, les assertions :

Mr. John Smith works for Kentek.
Kentek employs Mr. John Smith.

ont le même sens pour le système TULIP. Celui-ci sait qu'il y a une relation entre les expressions *works for* et *employs*.

La compréhension syntaxique

A ce niveau de compréhension, le système TULIP peut analyser une phrase et faire le lien entre des phrases qui ont le même sens, mais des structures syntaxiques différentes. Par exemple, les phrases :

Kentek employs Mr. John Smith.
Mr. John Smith is employed by Kentek.

fournissent la même information au système TULIP. La deuxième phrase est la forme passive de la première.

Le système comprend également des façons différentes d'exprimer une même relation de possession.

Par exemple :

the father of Jack
Jack's father

Le système TULIP peut également faire le lien entre une interrogation et une assertion lorsque celles-ci utilisent les mêmes expressions.
Exemple :

Kentek employs Mr. John Smith.
Who is Mr. John Smith employed by?

L'association de mots

Lorsqu'on pose au système TULIP une question qu'il ne peut analyser, le système trouve les assertions introduites précédemment qui contiennent le plus grand nombre de mots en commun avec la question. Ces assertions sont affichées comme réponses possibles à l'interrogation. Cette méthode simple permet assez souvent de trouver l'information désirée par l'utilisateur, bien qu'on puisse occasionnellement obtenir en même temps des informations non pertinentes.

Limites de la compréhension conceptuelle du système TULIP

Les fonctions de compréhension conceptuelle du système TULIP ont une capacité limitée. Au niveau syntaxique, le système peut comprendre la plupart des assertions introduites par l'utilisateur, mais non au niveau conceptuel. Par conséquent, il faut utiliser une terminologie cohérente. L'utilisateur peut accroître la compréhension conceptuelle du système en introduisant des règles qui définissent les relations entre les mots.

Procédures d'entrée et de sortie

Pour utiliser le système TULIP, il faut se trouver dans le répertoire :

/home/condor2/tulip

Charger TULIP en tapant :

tulip.bin

Vu la longueur du programme TULIP, le chargement du programme prend un certain temps. Lorsque le programme est chargé, le message de guidage Quintus Prolog suivant est affiché :

| ?-

Pour entrer dans le système TULIP, taper :

tulip.

Lorsque Tulip est prêt à recevoir les données de l'utilisateur, il affiche le message de guidage :

Enter a sentence :

Pour sortir du système Tulip, taper :

exit. (ou) quit.

Ceci fait afficher de nouveau le message de guidage Quintus Prolog :

| ?-

Pour sortir de Quintus Prolog, taper :

halt.

<u>Mise en application de la fonction d'aide du système Tulip</u>

Une fois à l'intérieur de Tulip, pour mettre en application la fonction d'aide du système Tulip, taper :

help.

Suspension de l'exécution du programme TULIP

L'utilisateur peut suspendre une exécution en cours du programme TULIP en tapant C (Control-C). Le message de guidage suivant est alors affiché :

Prolog interruption (h for help)?

Pour sortir du programme TULIP, l'utilisateur doit taper :

a

L'utilisateur revient alors au message de guidage Quintus Prolog. Il peut de nouveau entrer dans le programme Tulip en tapant :

tulip :

Données d'entrée pour le système Tulip

Les données d'entrée pour le système Tulip sont les suivantes :

[1] Les instructions - Une instruction est une ligne qui commence par un mot-clé (par exemple *read, save, restart*) et se termine par un point suivi d'un retour du curseur. Les instructions commandent au programme Tulip d'exécuter certaines opérations spéciales. Exemple :

restart.

[2] Les assertions - Une assertion est une ligne en anglais constituant une déclaration de fait et se terminant par un point suivi d'un retour du curseur. Tulip conserve les assertions en mémoire et les utilise pour répondre aux interrogations. Exemple d'assertion :

Bill is the husband of Mary.

Si le premier mot de la phrase n'est pas un nom propre, il n'est pas nécessaire que la première lettre soit une majuscule. Par exemple :

a cat is on the mat.

[3] Les interrogations - Une interrogation est une ligne en anglais ayant la forme d'une question terminée par un point d'interrogation suivi d'un retour du curseur. Lorsque l'utilisateur introduit une interrogation dans le système Tulip, celui-ci cherche une réponse en se basant sur l'information qui lui a été communiquée. Exemple d'interrogation :

who is Mary's husband?

Il n'est pas nécessaire de mettre une majuscule au premier mot d'une interrogation.

[4] Les règles - Une règle est une ligne en langage évolué de la forme :

<Conclusion> **si** <Prémisse1> {**et** <Prémisse 2>...).

où la <Conclusion> et les <Prémisses> sont des énoncés simples pouvant contenir certaines *variables* servant de substituts à des objets. Une variable est une lettre majuscule autre que *A* ou *I*. Les règles sont utilisées pour établir les relations entre les concepts. Exemple de règle :

X is the wife of Y if Y is the husband of X.

En utilisant cette règle et l'assertion suivante :

Bill is the husband of Mary.

le système Tulip peut alors répondre à la question :

Who is Bill's wife?

Le sens des verbes *have* et *is* dépend du type de complément du verbe. Par conséquent, on ne peut PAS utiliser une variable comme complément des verbes *have* et *is* :

[*] X has Y if X owns Y.
[*] X is a Y if the type of X is Y.

Toutefois, on peut utiliser une variable comme sujet des verbes *have* et *is* :

X has a window office if X is a manager.
X is a mammal if X has fur.

Une règle ne comportant qu'une seule variable peut souvent être exprimée sous forme d'assertion. Par exemple, les règles précédentes peuvent être exprimées sous les formes suivantes :

Every manager has a window office.
Everything that has fur is mammal.

[5) Les cartes professionnelles - Lorsqu'une ligne introduite ne se termine pas par un point ou un point d'interrogation, Tulip suppose qu'il s'agit de la première ligne d'une carte professionnelle. Si la

première ligne d'une carte professionnelle se termine par un point, il faut laisser un espace entre le point et le retour du curseur.

Tulip affiche le message de guidage suivant pour indiquer qu'il s'attend à ce qu'on introduise d'autres lignes pour la carte professionnelle :

I:

Une carte professionnelle doit se terminer par une ligne vierge.

Les instructions

save <nom> .

- sauvegarde des phrases tapées par l'utilisateur depuis la dernière instruction *Save* ou *Restart*. Les phrases sauvegardées sont identifiées par un <nom> , lequel est un mot alphanumérique. Si ce <nom> a été utilisé antérieurement avec une instruction *Save*, les phrases sont ajoutées à celles déjà identifiées par ce <nom>. On peut lire les phrases sauvegardées en utilisant l'instruction *Read*.

read <nom> .

- lecture des phrases sauvegardées à l'aide de l'instruction *Save <nom>* . Remarque : l'instruction *Save <nom>*. conserve les phrases dans un fichier (appelé *<nom> .text*) du répertoire **knowledge** de Tulip. On peut introduire un fichier de texte

dans le répertoire **knowledge** et le lire à l'aide de l'instruction *Read*. Ces fichiers peuvent être modifiés au moyen d'un programme d'édition.

restart.

- effacement du contenu de la mémoire de Tulip.debug.

debug.

- affichage du traitement d'une phrase par Tulip. Les données affichées sont très détaillées et ne sont probablement pas très utiles à un utilisateur moyen.

end debug.

- suppression de l'affichage détaillé du traitement des phrases.

help.

- assistance pour l'utilisation de Tulip.

Forme des assertions anglaises admissibles

En général, Tulip accepte les énoncés anglais qui sont des déclarations de fait.

Les énoncés sous forme négative ne sont pas admissibles. On ne peut introduire un énoncé comportant le mot *not*. Exemple :

[*] George is not a programmer.

Les énoncés comportant le mot *or* sont également inadmissibles. Exemple :

[*] Helen or Mary is married to Jim.

Une assertion doit commencer par un syntagme nominal sujet de l'énoncé.

Forme des interrogations anglaises admissibles

Les types de question *Wh* et *yes/no* sont tous les deux admissibles. Une question *Wh* est une question qui commence par un mot comme : *who, what, which, where ou when.* L'expression *How many* est également considérée comme un mot *Wh*. Un mot *Wh* peut être précédé d'une préposition. Exemples d'interrogations *Wh* :

Who is Frank's wife?
To whom is Frank married?
How many people work for Kentek?
Which employees live in Laval?

Notons que le mot *Wh* doit être le premier mot de la phrase (ou le deuxième s'il est le complément d'une préposition). L'interrogation :

[*] Jack is married to whom?

n'est donc pas admissible.

Pour la rendre admissible, il faudrait l'exprimer sous la forme :

To whom is Jack married?
ou
Whom is Jack married to?

Le mot *how* est inadmissible s'il est utilisé seul :

Jack runs quickly.
[*] How does Jack run?

Une question *yes/no* est une question à laquelle on peut répondre par *yes* ou *no*. Ces questions commencent ordinairement par le verbe *is* ou *does*. Exemples :

Does Frank work for Kentek?
Is Frank a manager?

Réponse aux interrogations par association de mots

Si l'on pose à Tulip une question qu'il ne peut analyser, il a recours à l'association de mots pour tenter d'obtenir des informations utiles. Par exemple, si l'on introduit des mots-clés suivis d'un point d'interrogation, Tulip affiche les assertions qui contiennent le plus grand nombre de mots-clés.

Les conjonctions

Le mot *and* ne peut être utilisé que de certaines façons. On peut l'utiliser entre des énoncés, par exemple :

Joe lives in Toronto and Henry lives in Montreal.

Si le sujet est le même dans les deux énoncés, il peut être omis dans le second :

Joe lives in Guelph and works in Toronto.

La conjonction *and* peut également être utilisée entre des clauses relatives (voir Syntagmes nominaux).

Les syntagmes nominaux

Un syntagme nominal peut être un syntagme nominal de nom commun, par exemple :

the tall man

ou un nom propre, par exemple :

John Doe

La première lettre d'un nom propre doit être une majuscule.

Un syntagme nominal de nom commun (mais non un nom propre) peut être modifié par une clause relative ou plusieurs clauses relatives jointes par des conjonctions. Par exemple, l'expression :

the man who lives in Halifax and is employed by Acme Inc.

constitue un syntagme nominal.

Les relatives réduites, comme par exemple :

[*] the man employed by Acme Inc.

ne sont pas admises par Tulip.

Celle-ci doit être présentée sous la forme :

the man who is employed by Acme Inc.

Un syntagme nominal de nom commun peut être singulier ou pluriel. Exemples :

the cat
the cats

Toutefois, les syntagmes nominaux reliés par une conjonction ne sont pas admissibles.

Par exemple :

[*] the men and women

[*] Bill and Jane

ne sont pas admissibles.

Les mots compris entre guillemets sont traités comme un nom propre. Exemple :

"The Young and the Restless" is a television program.

Les déterminants

Les syntagmes nominaux de noms communs sont généralement précédés d'un déterminant tel que *a, the, some, all* ou *every*. Les déterminants *most* et *few* ne sont pas admissibles.

Lorsqu'une assertion commence par le déterminant *a*, par exemple :

a bear has claws,

Tulip ne peut déterminer si le sens de l'assertion est :

some bear has claws ou every bear has claws.

Dans ce cas, Tulip affiche un menu où figurent ces deux options et demande à l'utilisateur de choisir le sens approprié.

Si le déterminant *a* est utilisé ailleurs qu'au début d'une phrase ou est utilisé dans le sens *some* au début d'une phrase, Tulip suppose qu'il est question d'un nouvel objet. Par contre, si le déterminant *the* est utilisé, Tulip suppose qu'il est question d'un objet mentionné dans une phrase précédente. Par exemple, si l'énoncé :

George has a car

a déjà été introduit et que l'utilisateur introduise l'énoncé :

Helen likes the car,

Tulip suppose que *the car* est la dernière voiture mentionnée, c'est-à-dire celle de George.

Par contre, si l'utilisateur introduit l'énoncé :

Helen likes a car,

Tulip ne suppose pas qu'Helen aime la voiture de George.

Le seul déterminant admissible devant un nom propre est *the* .

Exemples :

The Volvo

[*] a Volvo

Les pronoms

Tulip tente d'identifier les référents des pronoms singuliers à la troisième personne tels que *he*, *she*, *him*, *her* et *it*. Il tente également d'identifier le référent des pronoms possessifs singuliers à la troisième personne tels que *her* et *his*. Exemple :

Jack works for Acme Inc.
He owns a car.
Alice is his wife.
She gave him his coat.
who owns a car?
who is Jack's wife?
what did Alice give to Jack?

Dans l'exemple ci-dessus, Tulip suppose que le pronom *he* de la deuxième phrase désigne *Jack*, car il est la dernière personne de sexe masculin qui a été mentionnée.

Les référents des pronoms tels que *they*, *them* et *their* ne font PAS l'objet d'une tentative d'identification.

Les verbes

Les verbes au présent, au passé ou au futur sont admissibles. Toutefois, le temps des verbes n'est utilisé que pour interpréter les dates. Dans toute autre situation, Tulip ne tient pas compte du temps du verbe.

Les verbes modaux (ceux qui modifient d'autres verbes) tels que *can*, *should*, etc., sont également admissibles, mais sont négligés par le système.

Les phrases suivantes ont donc toutes le même sens pour Tulip :

Tweety can fly.
Tweety should fly.
Tweety could fly.
Tweety will fly.
Tweety flies.

À l'exception des verbes modaux, Tulip ne peut analyser les phrases contenant des verbes ayant d'autres verbes comme arguments (par exemple, persuader) :

[*] Tim persuaded Tom to hire Mary.

Les noms

Tulip peut parfois déterminer automatiquement quel type d'objet un nom propre désigne. Tulip connaît les prénoms masculins et féminins. Il utilise ses connaissances pour tenter de déterminer si les personnes mentionnées dans une phrase sont des hommes ou des femmes. De même, il suppose que les noms propres se terminant par *Limited*, *Incorporated* ou *Company* sont des sociétés. Il connaît également les noms de villes, de provinces et d'États.

Lorsqu'un nom apparaît dans une phrase, Tulip tente de l'associer à un objet qu'il connaît déjà. Il n'est pas nécessaire que ce nom soit absolument identique au nom déjà établi pour l'objet; toutefois, le nouveau nom doit être un sous-ensemble du nom établi. Par exemple, si Tulip connaît une personne appelée *Mr. Jack Smith*, lorsque le nom *Jack* ou *Mr. Smith* est utilisé, Tulip suppose qu'il s'agit de *Mr. Jack Smith*. Lorsqu'un nom peut désigner plus d'une personne connue de Tulip, ce dernier choisit celle qui a été mentionnée en dernier lieu. Ainsi, si *Mr. Jack Jones* a été mentionné plus récemment que *Mr. Jack Smith*, l'utilisation du nom *Jack* renvoie le système à *Mr. Jack Jones*. Lorsqu'une personne est mentionnée pour la première fois, on doit utiliser la forme la plus complète de son nom car cette personne ne pourra être désignée que par ce nom ou ses sous-ensembles par la suite. Un objet ne peut avoir plus d'un nom dans le système Tulip.

La date et l'heure

Voir la démonstration *dates* pour des exemples d'utilisation de la date et de l'heure par le système Tulip.

Les opérations arithmétiques

Voir la démonstration *cars* pour des exemples d'utilisation des termes *average, maximum, total* et *minimum* par le système Tulip.

Le messages d'erreur

Lorsque Tulip est incapable d'analyser une assertion ou une interrogation, il affiche le message suivant :

Syntactic parser failed towards the end of this segment :

Il affiche ensuite la partie de la phrase qu'il n'a pu analyser. Ordinairement, le dernier mot affiché dans la phrase est celui qui est la source du problème.

Lorsque Tulip peut analyser une interrogation, mais ne peut trouver une solution, il demande à l'utilisateur si celui-ci désire une liste des faits pouvant s'avérer utiles. Si l'utilisateur répond *Yes*, Tulip a recours à l'association de mots pour trouver les assertions ayant le plus grand nombre de mots en commun avec l'interrogation.

Les textes en français

Les capacités du système Tulip sont assez restreintes face aux textes en français. En général, Tulip ne peut analyser les phrases en français. Toutefois il peut utiliser l'appariement de mots avec certains mots français. Son vocabulaire français est limité aux mots pouvant apparaître sur une carte professionnelle et aux termes utilisés dans le domaine de l'intelligence artificielle.

Pour utiliser les accents et autres signes graphiques propres au français, il faut inclure les lignes suivantes dans le fichier *.cshrc* :

```
setenv DEFAULT_FONT "/home/condor/critter/fonts/f_screen.r.14"
setenv EMACS_FONT "/home/condor/critter/fonts/f_serif.r.14"
```

et avoir les lignes suivantes dans un fichier *.emacs_pro* dans le répertoire **home** :

```
(load "/home/condor/critter/.emacs_pro")
(load " /../isabelle/emacs/dead_key_accents")
```

Les signes graphiques peuvent être ajoutés aux lettres en utilisant les touches R1 à R5 qui se trouvent à droite des touches principales sur une machine Sun 3. On ajoute un signe graphique en enfonçant la touche R correspondante avant d'enfoncer la touche de la lettre à laquelle le signe doit être ajouté.

Les signes graphiques des touches R sont :

R1 - accent aigu
R2 - accent grave
R3 - cédille
R4 - tréma
R5 - accent circonflexe

L'expression des connaissances dans le système Tulip

Tulip peut être utilisé pour exprimer la transmission de propriété entre des classes d'objets. Par exemple, on peut introduire dans le système les assertions :

every bird can fly.
every robin is a bird.
Ernie is a robin.

Si l'on pose alors la question

Can Ernie fly?

Tulip répond *Yes*. Étant donné que l'utilisateur a déclaré que chaque rouge-gorge est un oiseau, les propriétés associées aux oiseaux (par exemple, la capacité de voler) sont transmises à tous les rouges-gorges.